

## REMARKS

Applicants respectfully traverse and request reconsideration.

Claims 1, 4, 5, 11 and 12 stand rejected under 35 U.S.C. §102(e) as being anticipated by Chen et al. Applicants respectfully request reconsideration and respectfully submit that the Chen reference does not teach the claimed method and apparatus for several reasons. In the “Response to Arguments” section of the Office Action, the Office Action attempted to address Applicants’ arguments, but does not appear to take into the account the operation, as best understood, of the Chen system. The Chen reference describes a system and method of installing application files and corresponding setup configurations for a plurality of mobile devices from a storage source. The setup package files are predetermined setup package files that are different for each type of mobile device. The setup package includes a single file having a first portion that includes the application setup instructions and a second portion that includes the application files. (Col. 3, lines 4-7). These predetermined setup package files contain all the files and user settings for installing and running the program on a mobile device. (Col. 4, lines 62-64). As noted in the reference, these setup package files are not dynamically constructed from a set of code modules based on the actual system configuration parameters, but instead are predetermined and pregenerated such that several setup package files are created for each of different configuration of mobile devices. Although the reference uses the words “dynamic creation of the setup package file 10A on the mobile device 3 is not ideal, since the mobile device 3 operates with less memory than the desktop computer 4.” (Col. 10, lines 8-11). The “dynamic creation” referred to in the cited portion refers to the fact that the setup package files are pregenerated and downloaded by the mobile device. As the Chen reference describes, once the mobile device is connected to the desktop computer, the setup package file is created “on” -- i.e. downloaded to --

the mobile device 3. Moreover, for example, in Col. 8, the generator module that generates the setup package files is the desktop computer. As such, the reference in Col. 10 of the “dynamic creation” on the mobile device refers to the downloading of the pregenerated setup package file by the mobile device since the actual generation of the setup package file occurs on the desktop device. As such, Chen does not teach dynamically bundling a software package in response to a system type being known, but instead teaches downloading an already created setup package file which are already created and wherein a separate package file exists for each different type of mobile device. Accordingly, Chen fails to teach, among other things, obtaining an actual system configuration parameter and dynamically constructing at least one code bundle from a set of code bundle modules based on the actual system configuration parameter as there are no code bundles dynamically constructed by the Chen system, but to the contrary different generated setup files, which are stored in desktop computers are downloaded based on the type of mobile device. As such, Applicants respectfully submit that these claims are in condition for allowance. In addition, Applicants, only for the sake of argument, assume that the files and applications as described could be considered code modules as described.

Claims 2 and 9 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Chen et al. in view of Biggs et al. As to these claims, the Office Action agrees that the Chen reference does not teach, among other things, to dynamically construct a code bundle for every driver entry point associated with a software driver. The Office Action alleges that it would have been obvious to one of ordinary skill in the art at the time of the invention to construct an executable program and memory as taught by Chen and dynamically constructing a code bundle for every driver entry point associated with the software driver as taught by Biggs since Biggs mentions software drivers. However, Applicants respectfully note that upon the reading of the Chen

reference and Biggs reference, as noted above, the Chen reference does not in fact teach dynamically creating any software bundle based on obtaining an actual system configuration parameter. Moreover, the actual program referred to in the Chen reference is a file or application program that is to be installed on a mobile device and application setup instructions are included with the application files. As such, Chen only refers to applications programs that are already generated and are complete application programs when they are received by the mobile device.

The Biggs reference is directed to a video driver system which allows a single video driver to be used to drive any one of a plurality of different controllers in any one of a plurality of colored depths. As noted, the claim requires dynamically constructing a code bundle for every driver entry point associated with a software driver. As such, a code bundle, constructed from code modules based on actual system configuration parameters must be constructed for every driver entry point associated with a software driver. However, as noted, the Chen reference is not directed to constructing dynamically, code bundles for software drivers nor is the Biggs reference in any way directed to constructing code bundles dynamically, for software drivers. In fact, it appears that the Biggs reference and Chen reference teach an opposite approach to that claimed by Applicants. The Biggs reference teaches storing a vector table containing a plurality of device specific routines. Each entry in the vector table points to a device specific routine for hardware acceleration of a primitive command or to a default routine which sets the video control in a specific mode which can be operated by a set of colored depth specific routines. As such, there does not appear to be any discussion of constructing dynamically, a code bundle for every driver entry point associated with a software driver in either of the references. Accordingly, these claims are also in condition for allowance. If the

rejection is maintained, Applicants respectfully request a showing as to where Biggs teaches dynamically constructing a code bundle for every entry point of address.

Claims 3 and 10 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Chen. These claims require, among other things, that dynamically constructing a code bundle includes adding at least a jump instruction and call instruction for every code bundle so that either a call or jump instruction is added as part of the dynamic construction of the code bundle. The Office Action states and admits that the Chen reference does not teach dynamically constructing the code bundle with a jump or call instruction. The Office Action goes on to state that hence it must somehow be inherent in the Chen reference since the alleged code bundle generation in Chen would require the use of noncontiguous memory space to access different code modules that exist in the bundle. However, the Chen reference would not need such a jump or call routine since separate setup package files which contain setup instructions in associated application files are stored for different mobile devices. The setup package files that are generated are likely stored as “files” and as such would not include jump commands. If this rejection is maintained, Applicants respectfully request factual support for the official notice taken in the Office Action. Since the cited reference does not appear to teach such claimed invention, Applicants respectfully submit that the claims are in condition for allowance.

Claims 6 and 13 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Chen in view of Amberg. As to Claim 6, both the Chen and Amber references fail to teach “wherein the step of dynamically constructing at least one code bundle includes: in response to storing dynamic configuration parameters, using indexed code modules associated with the stored dynamic configuration parameters to determine which code modules are selected to define a portion of a software driver.” Because neither the Chen nor the Amberg reference teaches all the

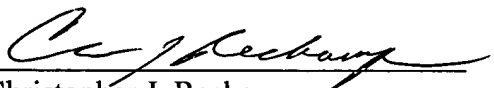
claim limitations of Claim 6, and because Claim 6 relies upon an allowable base claim, Claim 6 is believed to be allowable.

As to Claim 13, the Applicants respectfully restate the relevant remarks with respect to Claim 5. Applicants also add that Claim 13 is dependent upon an allowable base claim and contains further patentable material. Applicants believe Claim 13 to be allowable.

Claims 7 and 14 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Chen in view of Reha. As to Claims 7 and 14, the Applicants respectfully restate the relevant remarks with respect to Claim 1. Applicants also add that Claims 7 and 14 are dependent upon an allowable base claim and contain further patentable material. Applicants believe Claims 7 and 14 to be allowable.

Applicants respectfully submit that the claims are in condition for allowance and respectfully request that a timely Notice of Allowance be issued in this case. The Examiner is invited to contact the below listed attorney if the Examiner believes that a telephone conference will advance the prosecution of this application.

Respectfully submitted,

By:   
Christopher J. Reckamp  
Reg. No. 34,414

Dated: December 2, 2003

Vedder, Price, Kaufman & Kammholz, P.C.  
222 North LaSalle Street  
Chicago, Illinois 60601  
Telephone: (312) 609-7599  
Facsimile: (312) 609-5005  
Email: creckamp@vedderprice.com